Thread

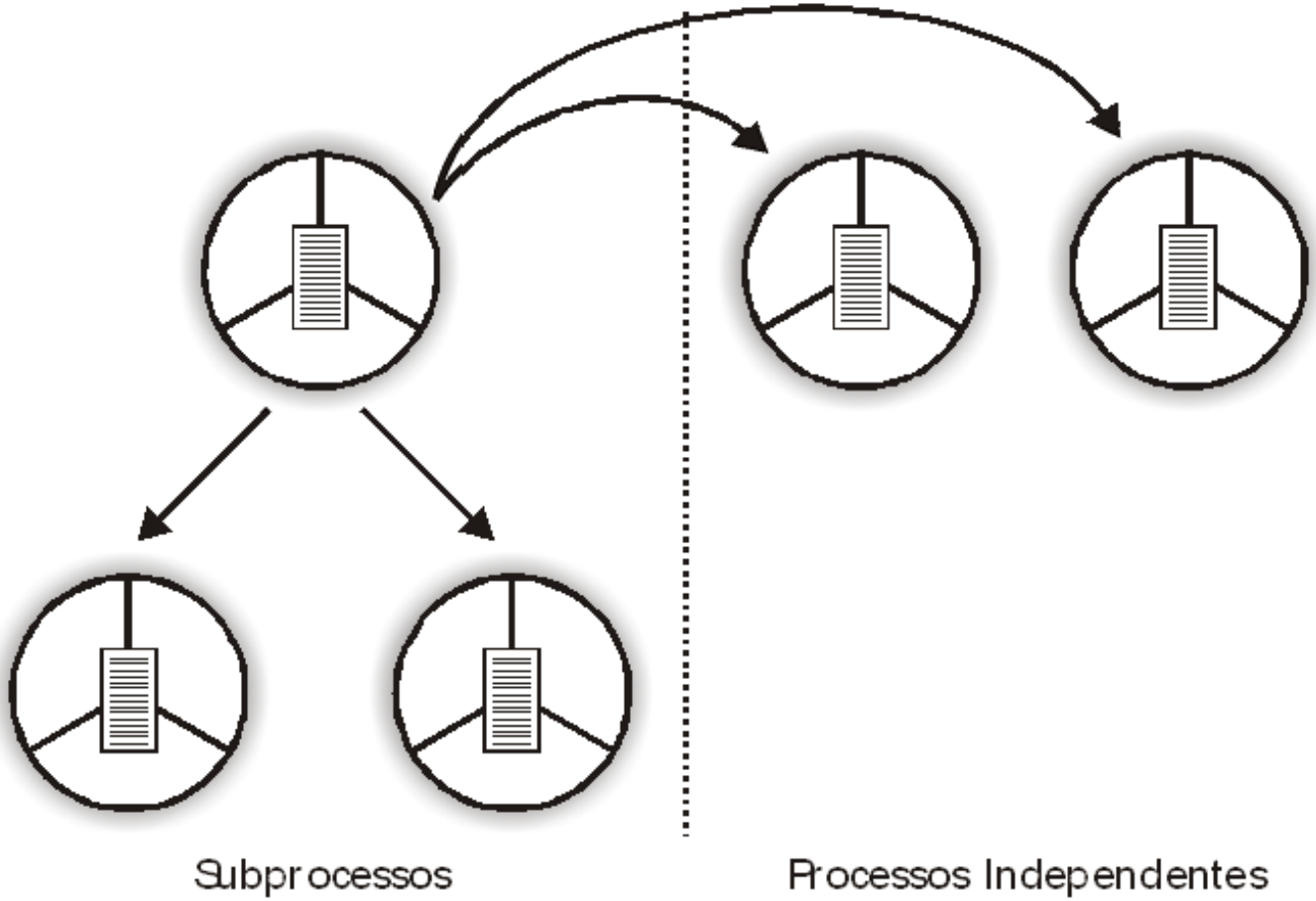
Introdução

- Thread
  - Sub-rotina que pode ser executada assincronamente
  - Programador especifica os threads
  - SO garante o ambiente de execução
  - Técnica utilizada a partir da década de 1980
  - Programas concorrentes
  - Maioria dos SO's atuais suportam

Ambiente Monothread

- Processos com apenas um programa por espaço de endereçamento
- Concorrência
  - Processos independentes e Subprocessos
- Para cada processo Usado
  - Sobrecarga de criação e destruição
  - Comunicação lenta
- Exemplos:
    - MS-DOS; Versões iniciais do MS-Windows

Ambiente Monothread

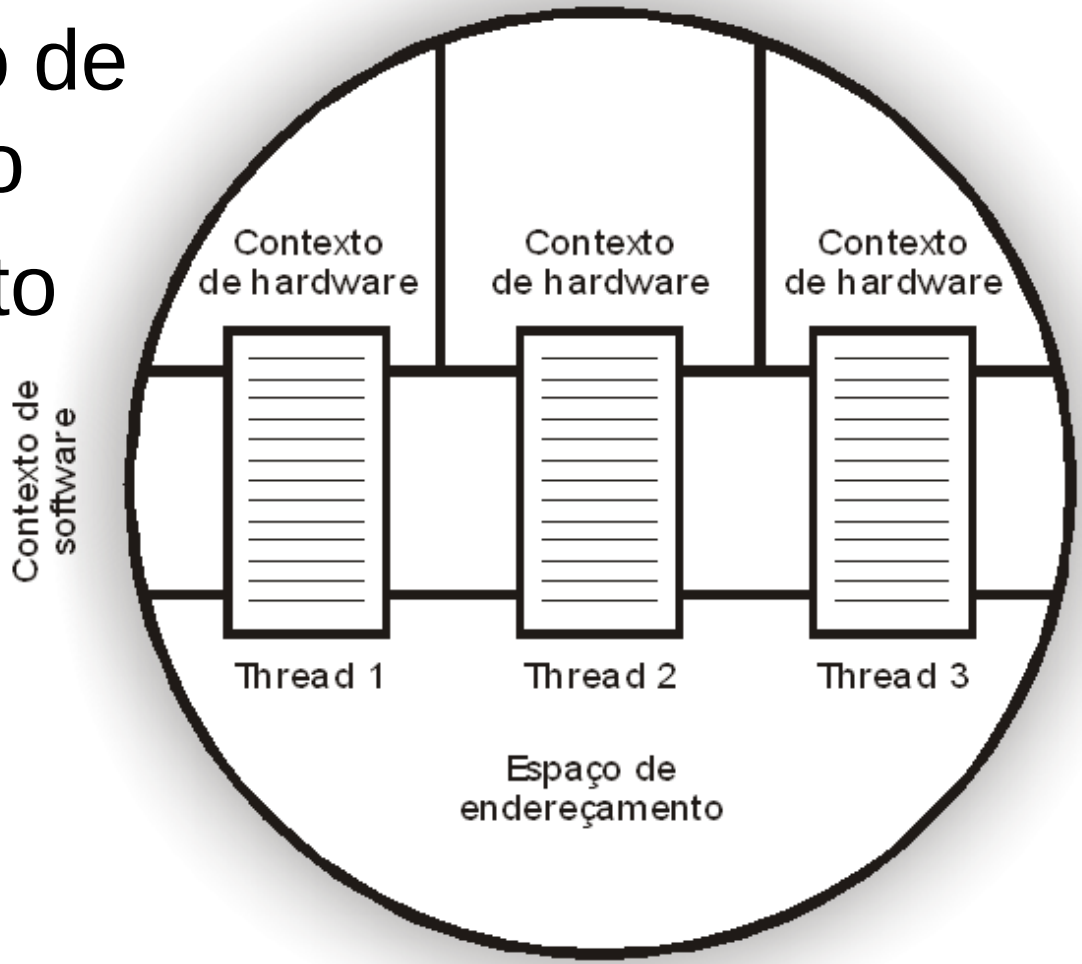


Ambiente Multithread

- Programas associados a threads
- Compartilhamento do espaço de endereçamento
  - Comunicação rápida
  - Necessidade de manter integridade dos dados
    - Mecanismos de sincronização

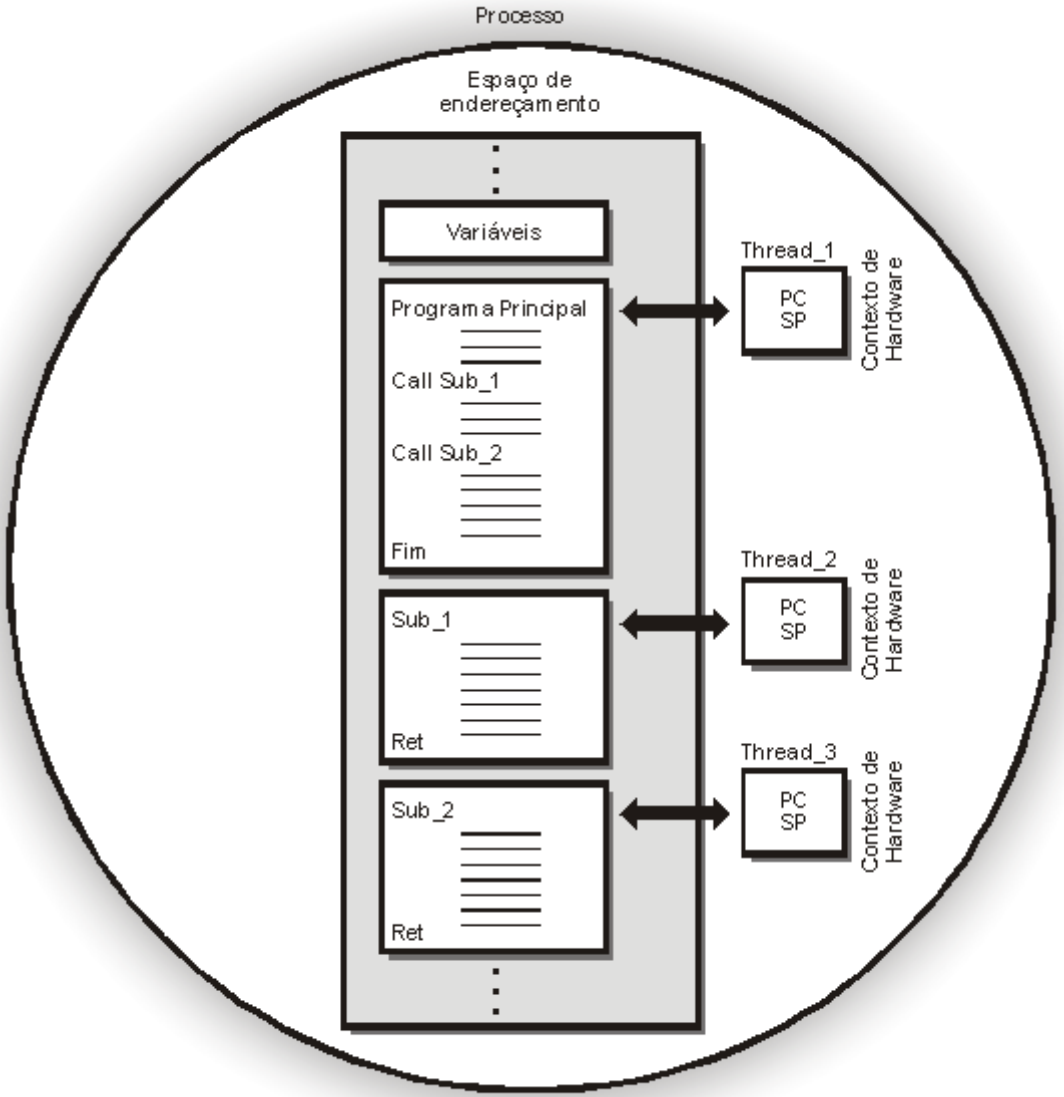
Processo e Threads

- Mesmo espaço de endereçamento
- Mesmo contexto de software
- Contexto de Hardware diferente



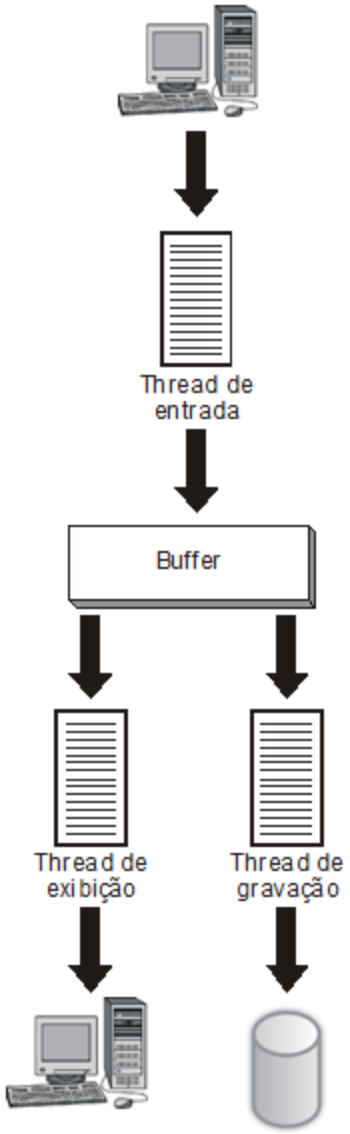
Aplicação Multithread

- O Programador especifica;
- O Ambiente de programação especifica



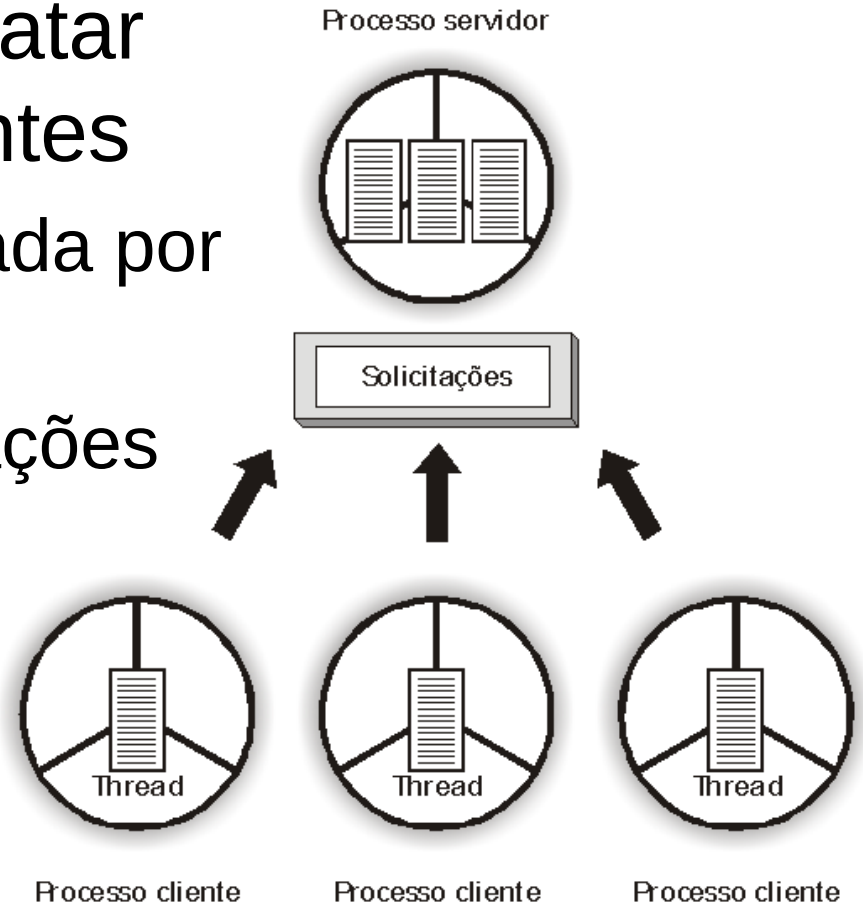
Aplicações de Threads

- Melhor utilização de recursos
  - Cada solicitação de tratamento de hardware é realizado por um thread
  - Algumas tarefas podem ser realizadas em background enquanto operações E/S são realizadas.



Aplicações de Threads

- Servidores precisam tratar as solicitações de clientes
  - Cada solicitação é tratada por uma thread.
  - Assim, diversas solicitações são tratadas "simultaneamente"

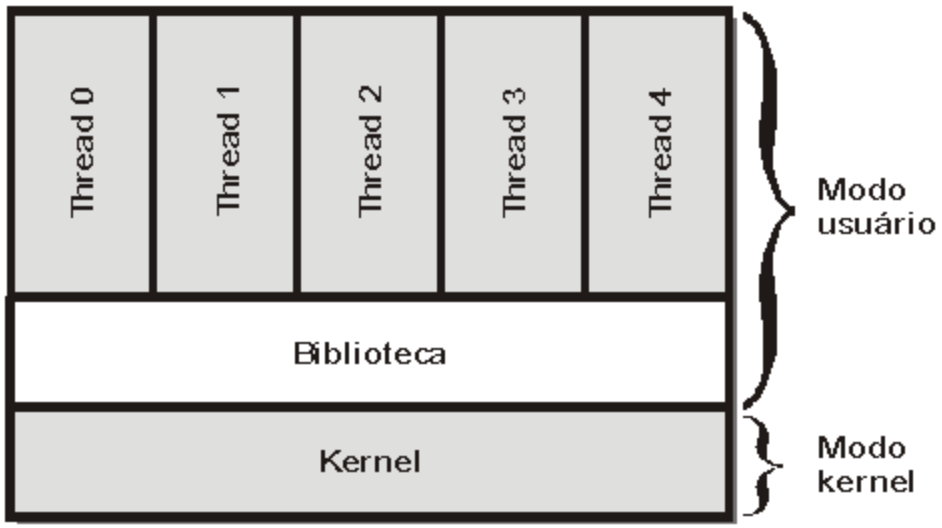


Threads – Implementação

- Pacote de Threads
  - Conjunto de rotinas usadas para manipular threads.
- Diferentes Abordagens
  - Modo Usuário
  - Modo *Kernel*
  - Modo Híbrido
  - Modo *Scheduler Activations*

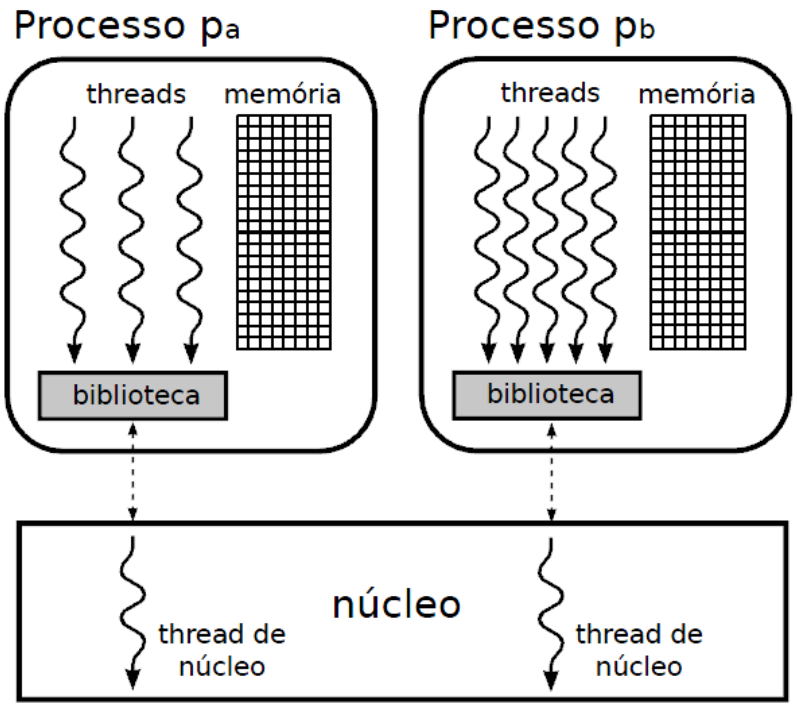
Threads – Modo Usuário

- Gerenciados pela aplicação
  - S.O. não conhece
  - Limitados ao processo pai
    - Mudança de Estado
    - Sinais



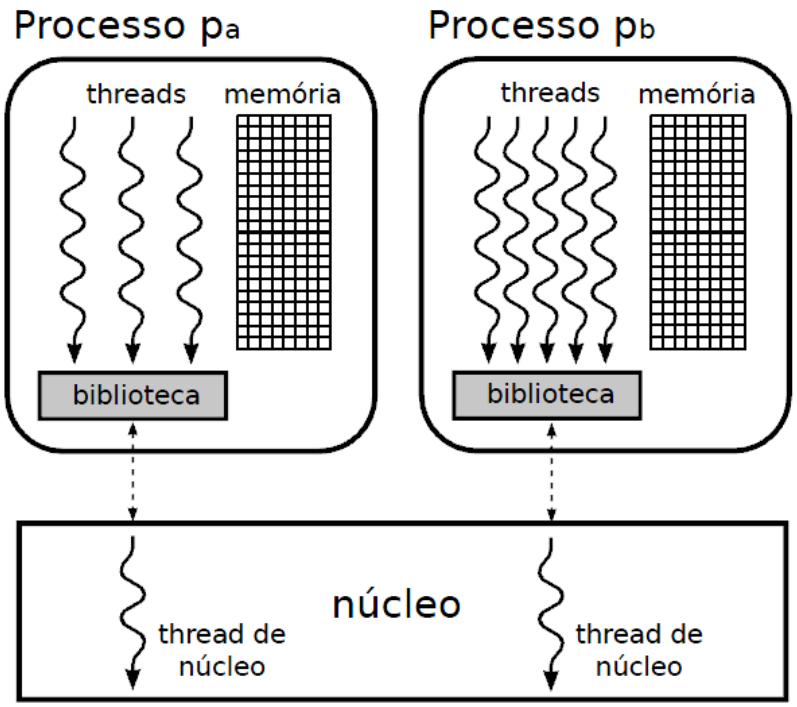
Threads – Modo Usuário (N:1)

- Vantagens:
  - Fácil implementação
  - Pouca sobrecarga ao núcleo.
  - Útil para aplicações com milhares de threads
    - Simulação de cidades, jogos, etc.



Threads – Modo Usuário (N:1)

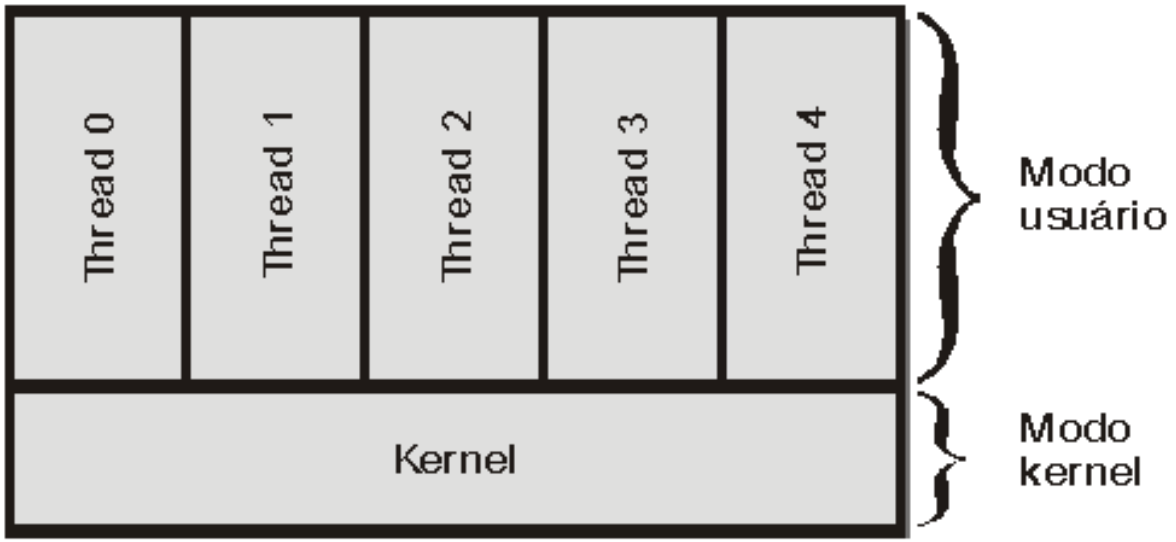
- Desvantagens:
  - Interação com hardware depende do núcleo
    - Todo o processo é suspenso.
    - Concorrência é prejudicada.



  - Escalonamento de threads ineficiente.
    - Depende de escalonamento do processo.

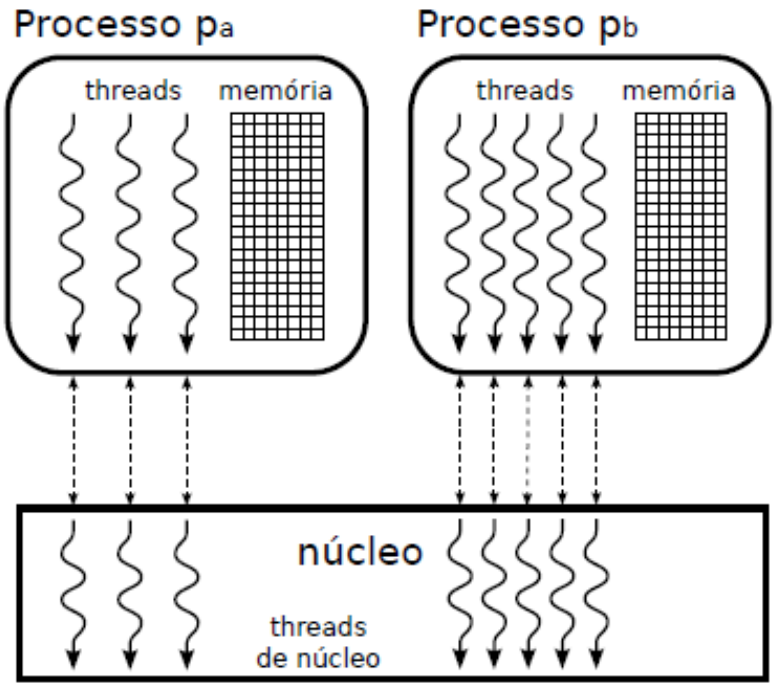
Threads – Modo Kernel (1:1)

- Gerenciados pelo Kernel do S.O.
  - Disponibilizados por *Chamadas de Sistemas*.
  - Thread é a unidade de escalonamento



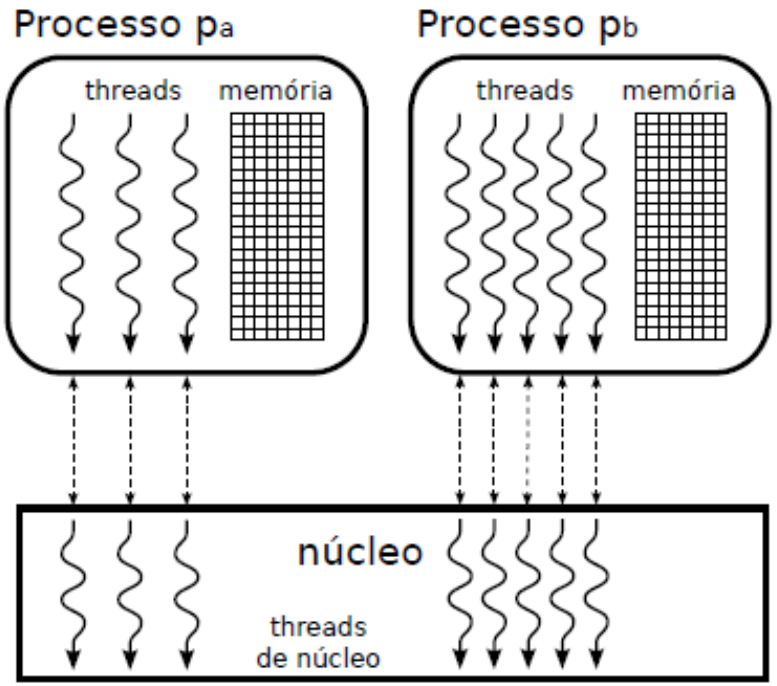
Threads – Modo Kernel (1:1)

- Vantagens:
  - Alocação em diversos processadores.
  - Escalonamento controlado pelo SO.
  - E/S não causa suspensão de todas os threads do processo.



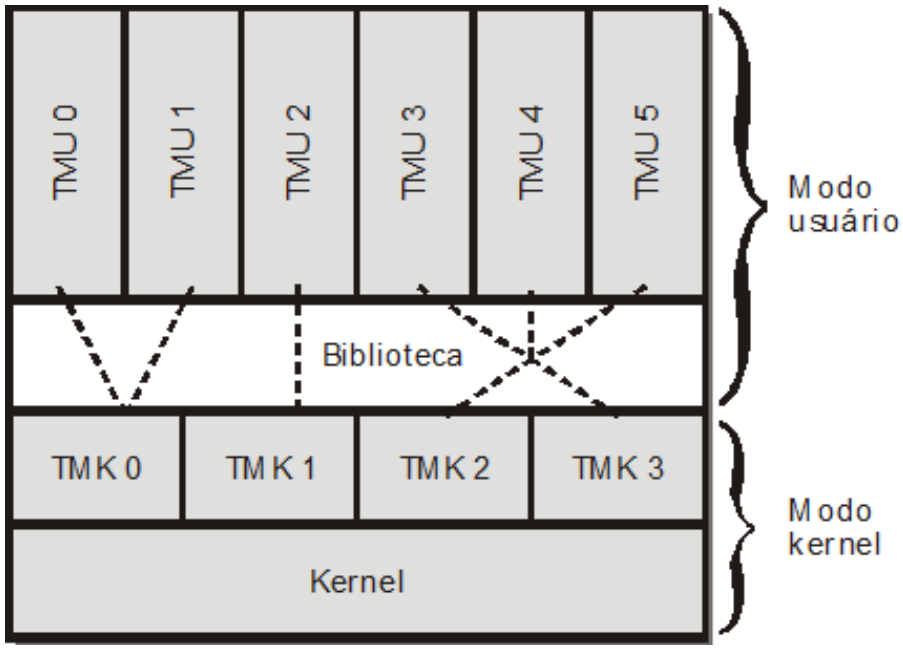
Threads – Modo Kernel (1:1)

- Desvantagens
  - Pouco escalável
    - Overhead de gerenciamento para o Sistema Operacional.
  - Mudanças de nível usuário para nível supervisor
    - Maior overhead de mudança de modo de acesso



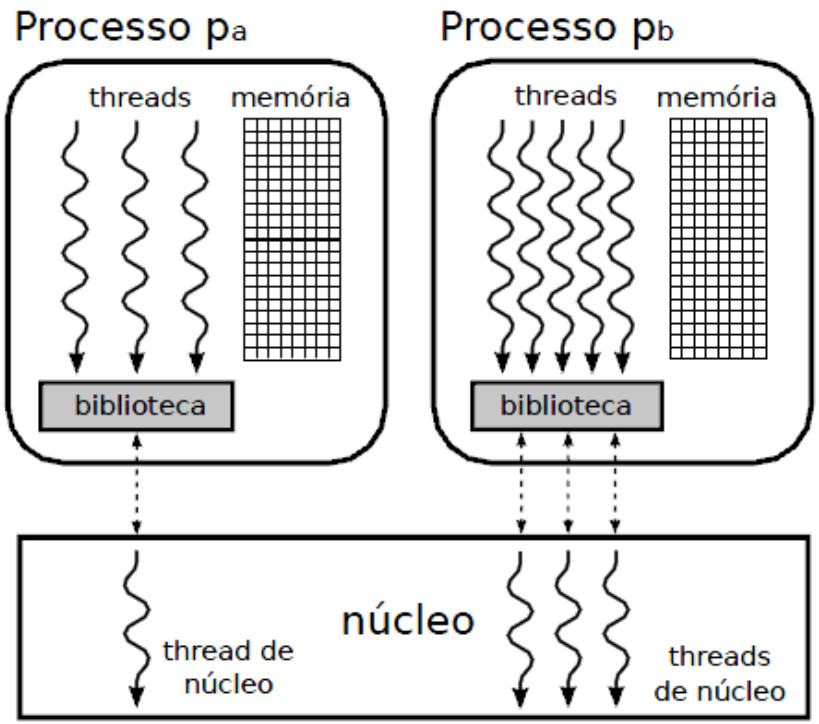
Threads – Modo Híbrido (N:M)

- Reúne características de TMU e TMK
- Utiliza duas camadas de Threads
  - Uma do Kernel
  - Uma da aplicação



Threads – Modo Híbrido (N:M)

- Vantagens:
  - Tenta unir o melhor dos dois mundos.
- Desvantagens:
  - Complexa implementação
  - Overhead de gerenciamento do SO



Threads: Resumo Implementação

Modelo	**N:1**	**1:1**	**N:M**
Resumo	Todos os N *threads* do processo são mapeados em um único *thread* de núcleo	Cada *thread* do processo tem um *thread* correspondente no núcleo	Os N *threads* do processo são mapeados em um conjunto de M *threads* de núcleo
Local da implementação	bibliotecas no nível usuário	dentro do núcleo	em ambos
Complexidade	baixa	média	alta
Custo de gerência para o núcleo	nulo	médio	alto
Escalabilidade	alta	baixa	alta
Suporte a vários processadores	não	sim	sim
Velocidade das trocas de contexto entre *threads*	rápida	lenta	rápida entre *threads* no mesmo processo, lenta entre *threads* de processos distintos
Divisão de recursos entre tarefas	injusta	justa	variável, pois o mapeamento *thread*→ processador é dinâmico
Exemplos	GNU Portable Threads	Windows XP, Linux	Solaris, FreeBSD KSE

Exercícios

1. Explique a diferença entre unidade de alocação de recursos e unidade de escalonamento?

2. Quais as vantagens e desvantagens do compartilhamento do espaço de endereçamento entre threads de um mesmo processo?

3. Como o uso de threads pode melhorar o desempenho de aplicações paralelas em ambientes com múltiplos processadores?